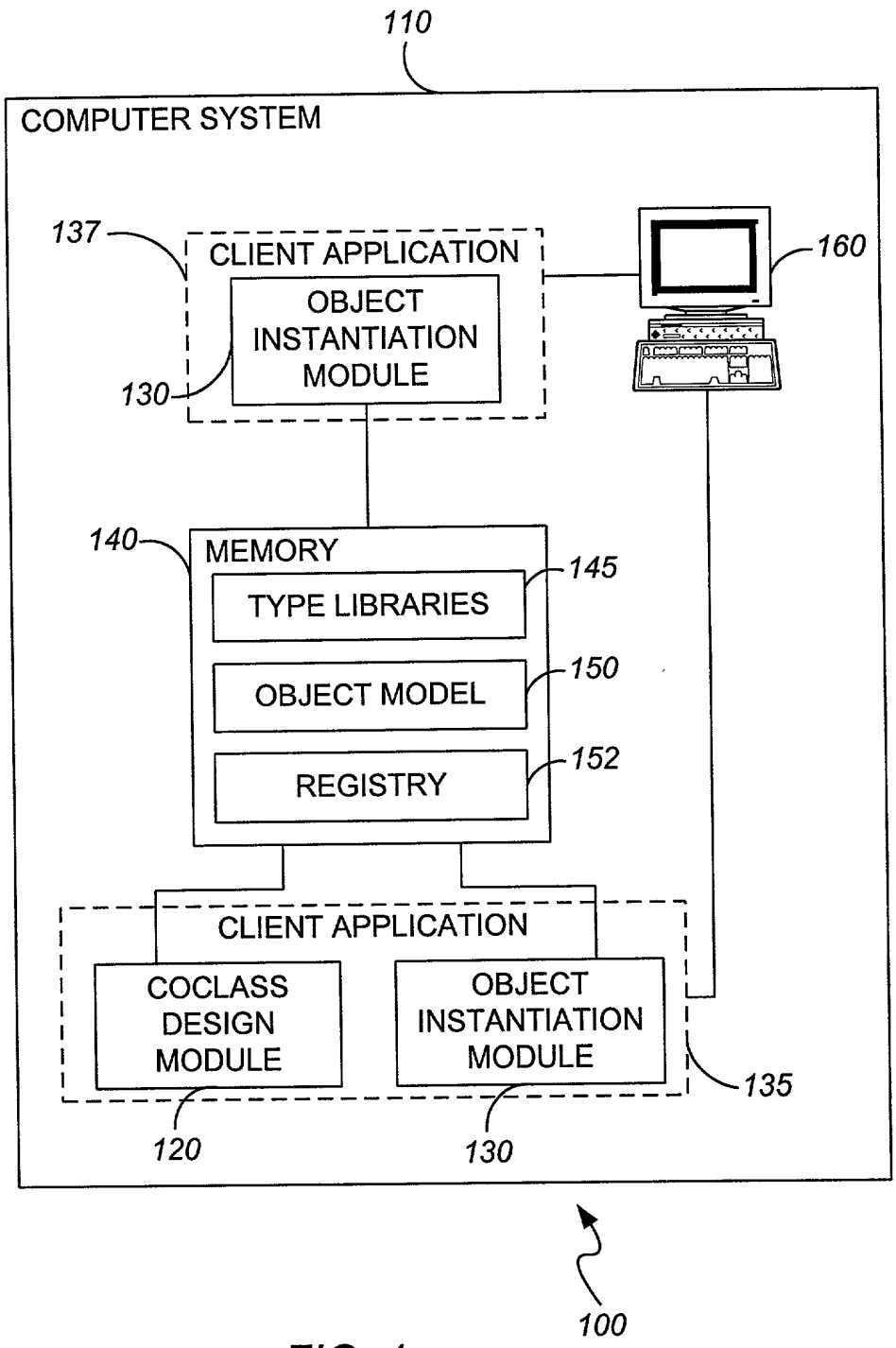

110
COMPUTER SYSTEM
137
CLIENT APPLICATION
OBJECT INSTANTIATION MODULE
130
160
140
MEMORY
145
TYPE LIBRARIES
150
OBJECT MODEL
152
REGISTRY
CLIENT APPLICATION
COCLASS DESIGN MODULE
OBJECT INSTANTIATION MODULE
135
120
130
100

FIG. 1

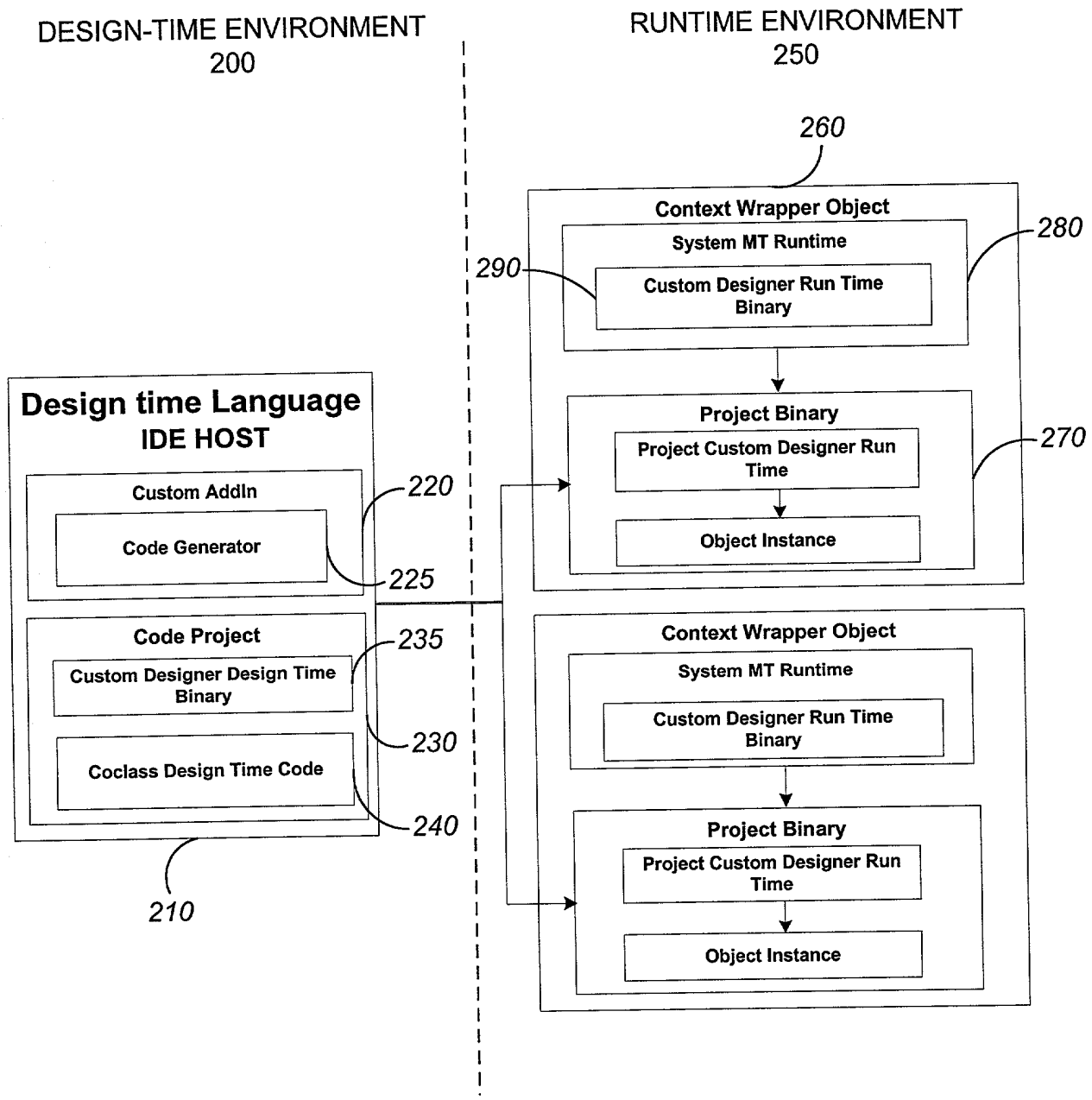

DESIGN-TIME ENVIRONMENT
200
RUNTIME ENVIRONMENT
250
260
Context Wrapper Object
System MT Runtime
Custom Designer Run Time Binary
280
290
Project Binary
Project Custom Designer Run Time
Object Instance
270
Design time Language
IDE HOST
Custom AddIn
Code Generator
220
225
Code Project
Custom Designer Design Time Binary
235
230
Coclass Design Time Code
240
210
Context Wrapper Object
System MT Runtime
Custom Designer Run Time Binary
Project Binary
Project Custom Designer Run Time
Object Instance

FIG. 2

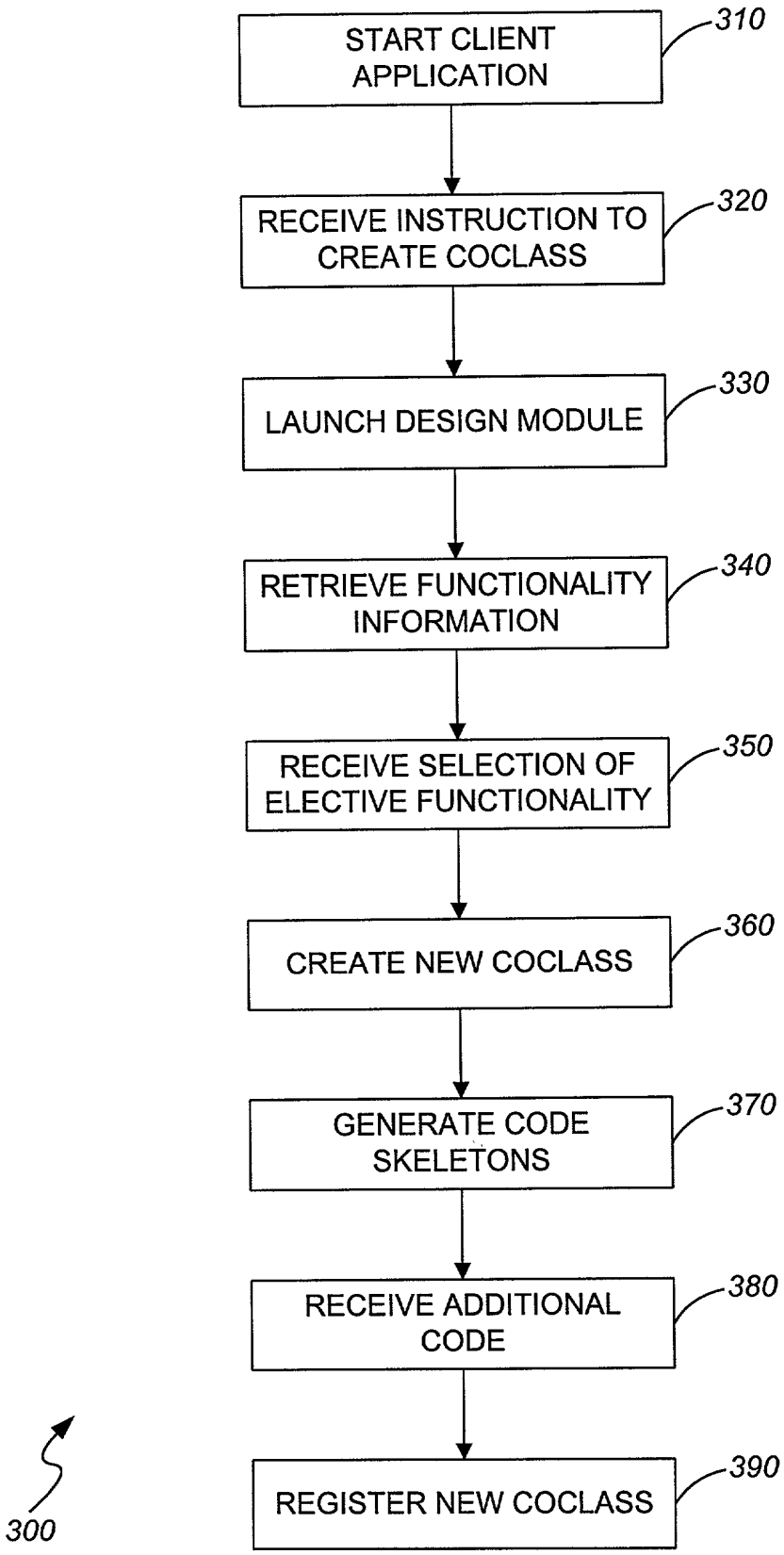

START CLIENT APPLICATION
310
RECEIVE INSTRUCTION TO CREATE COCLASS
320
LAUNCH DESIGN MODULE
330
RETRIEVE FUNCTIONALITY INFORMATION
340
RECEIVE SELECTION OF ELECTIVE FUNCTIONALITY
350
CREATE NEW COCLASS
360
GENERATE CODE SKELETONS
370
RECEIVE ADDITIONAL CODE
380
REGISTER NEW COCLASS
390
300

FIG. 3

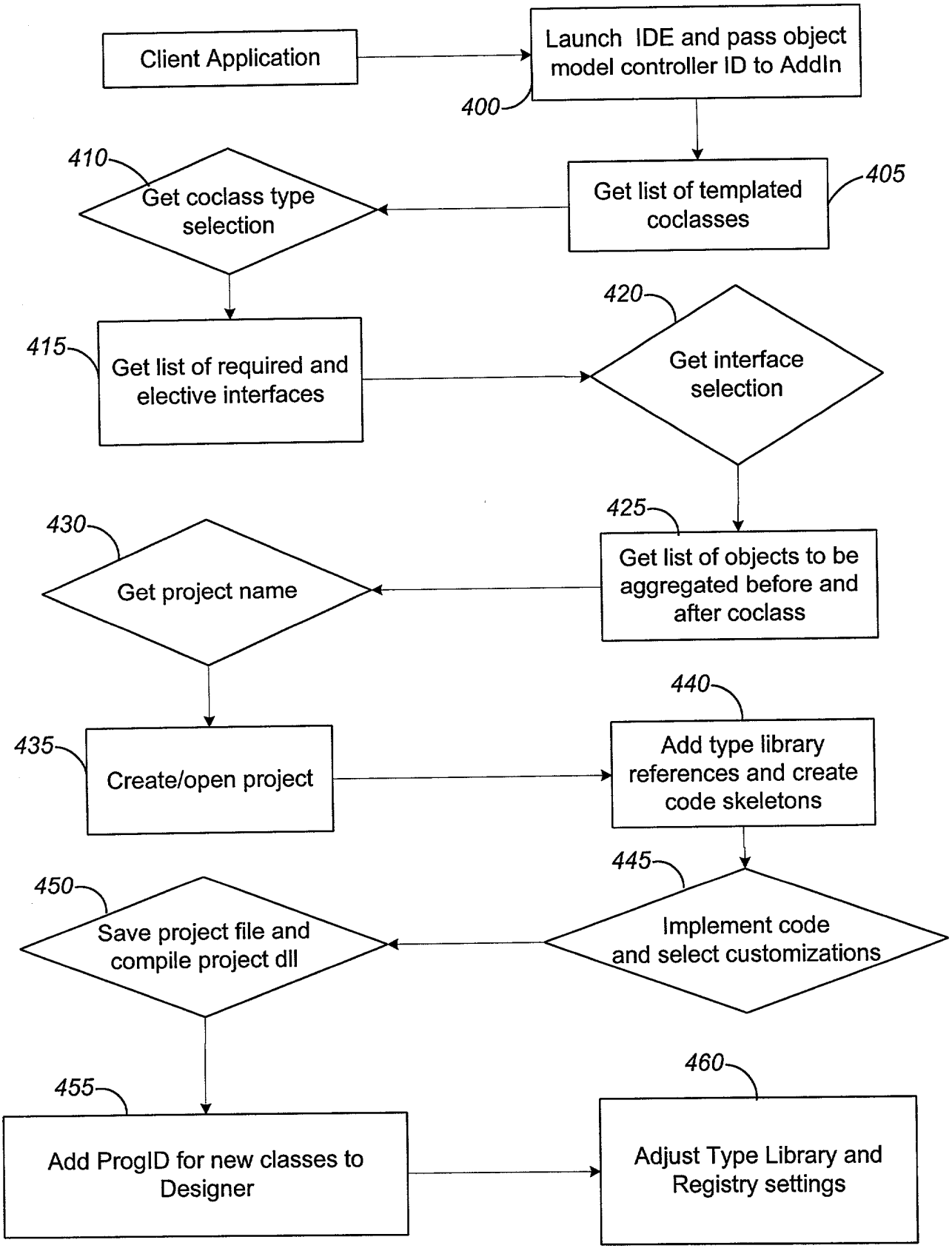

Client Application
Launch IDE and pass object model controller ID to AddIn
400
Get list of templated coclasses
405
410
Get coclass type selection
415
Get list of required and elective interfaces
420
Get interface selection
425
Get list of objects to be aggregated before and after coclass
430
Get project name
435
Create/open project
440
Add type library references and create code skeletons
445
Implement code and select customizations
450
Save project file and compile project dll
455
Add ProgID for new classes to Designer
460
Adjust Type Library and Registry settings

FIG. 4A

VBA Project Instantiation-1

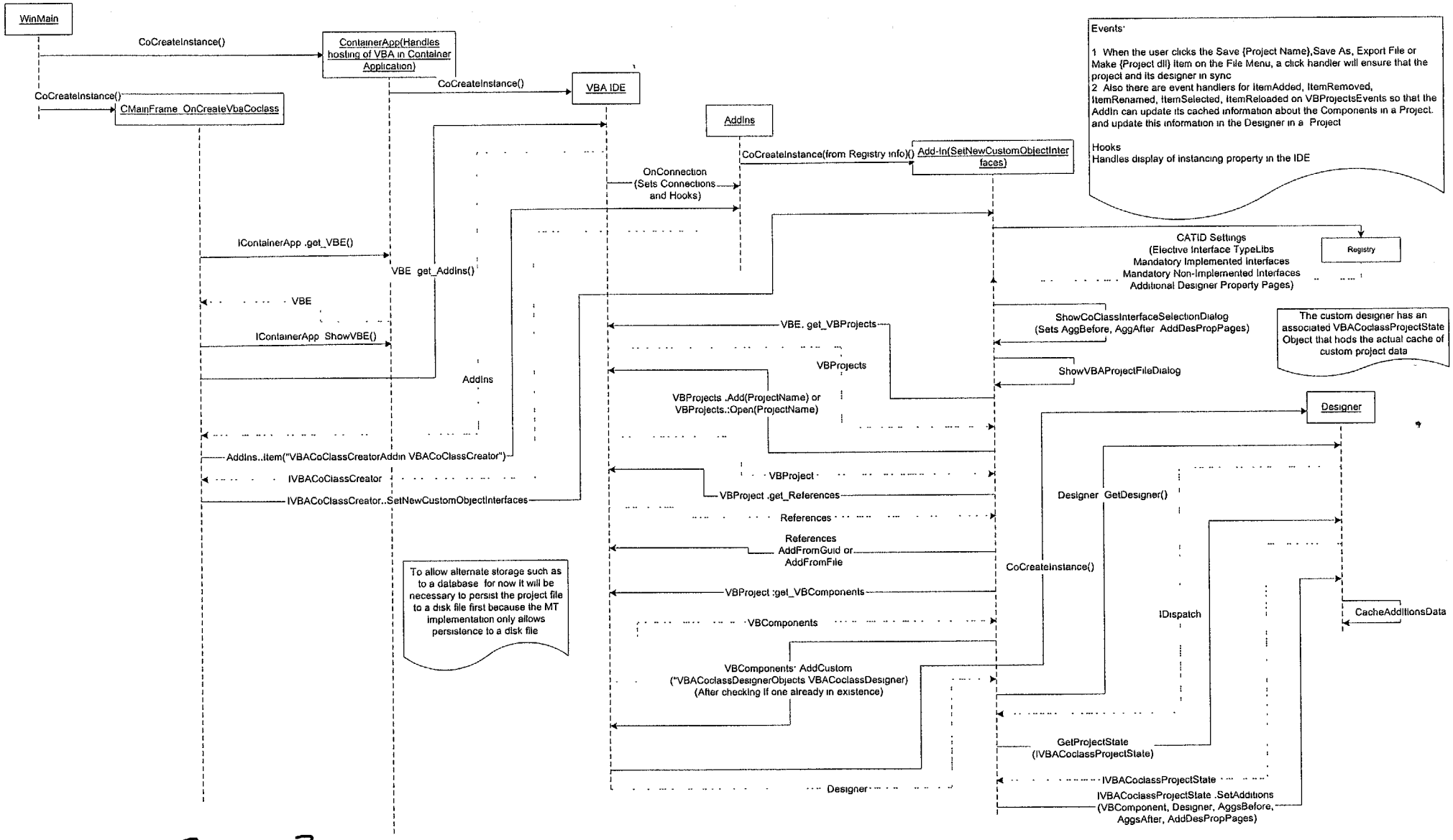


FIG. 4B

VBA Project Instantiation-2

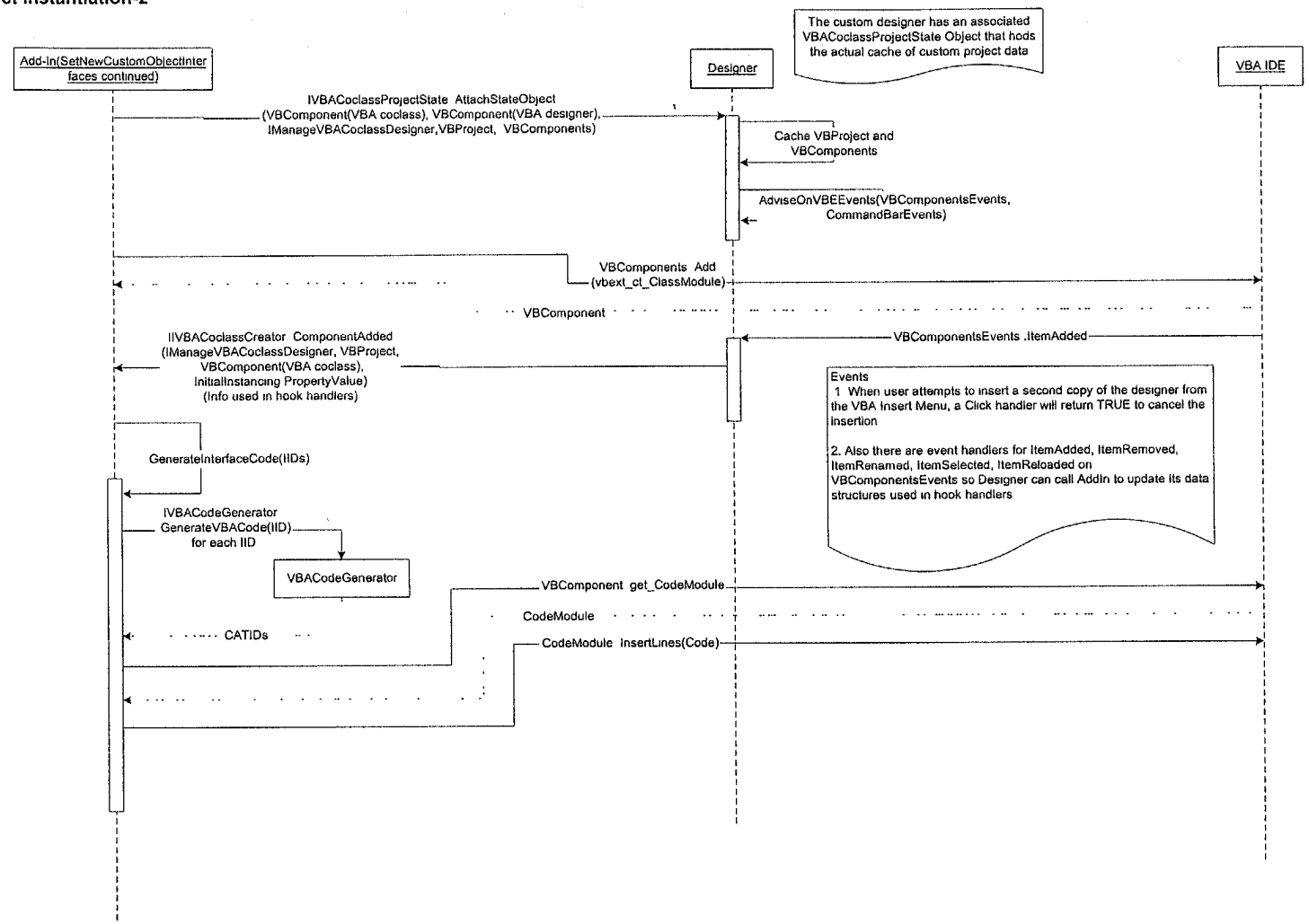


FIG. 4C

VBA Project Compilation & Save

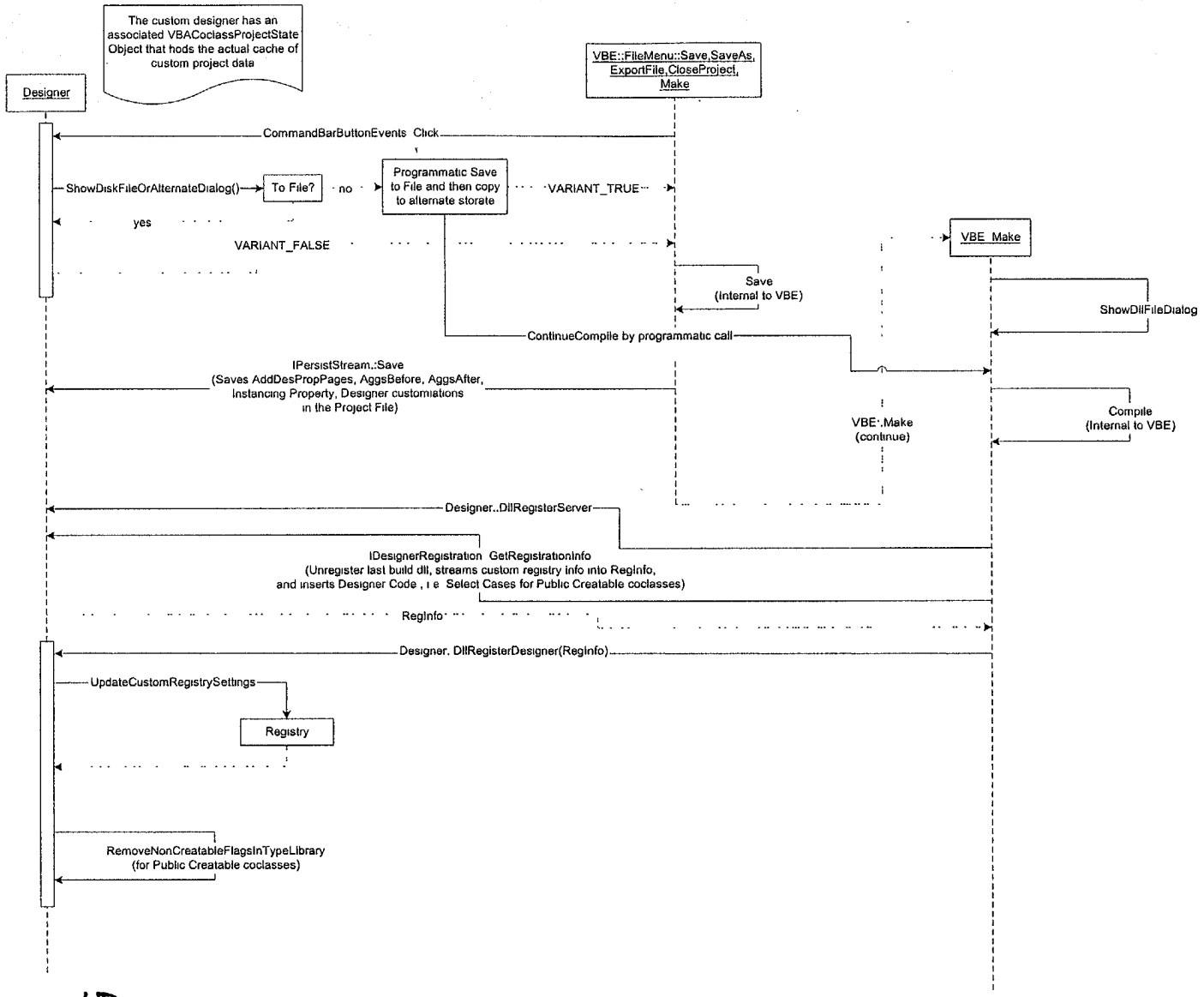
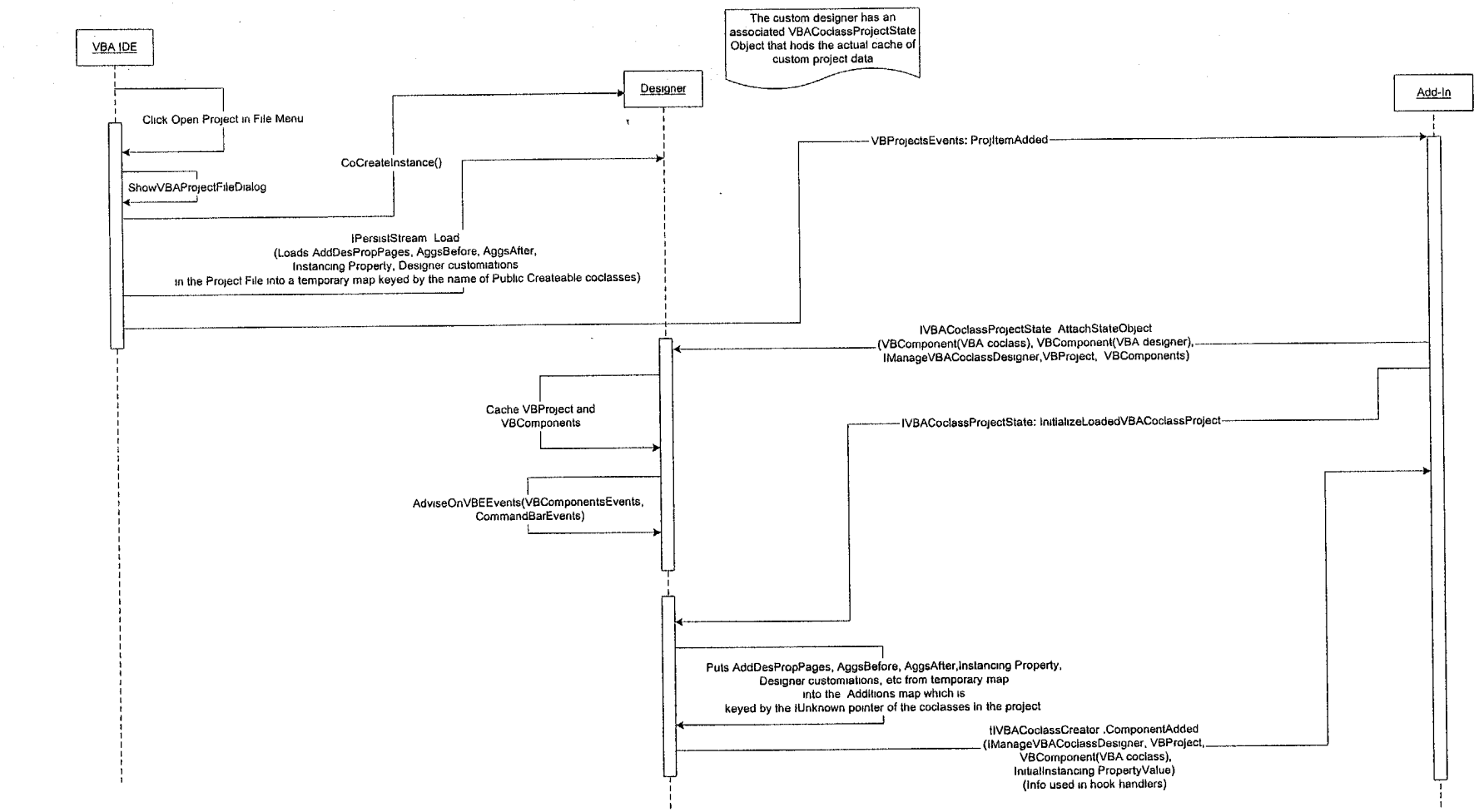


FIG. 4D

VBA Project Open, Removal



When the user selects deleting a project from the VBA IDE, the VBProjectsEvents. ProjItemRemoved handler in the AddIn removes the project's custom settings in the data structures in the AddIn Also when the user inserts/deletes a designer into/ from a project, or inserts/deletes a class into/from a project, the VBCommandBarEvents'::Click event handler in the AddIn and designer, respectively, add to or delete from the data caches in the AddIn and designer

Fig. 4E

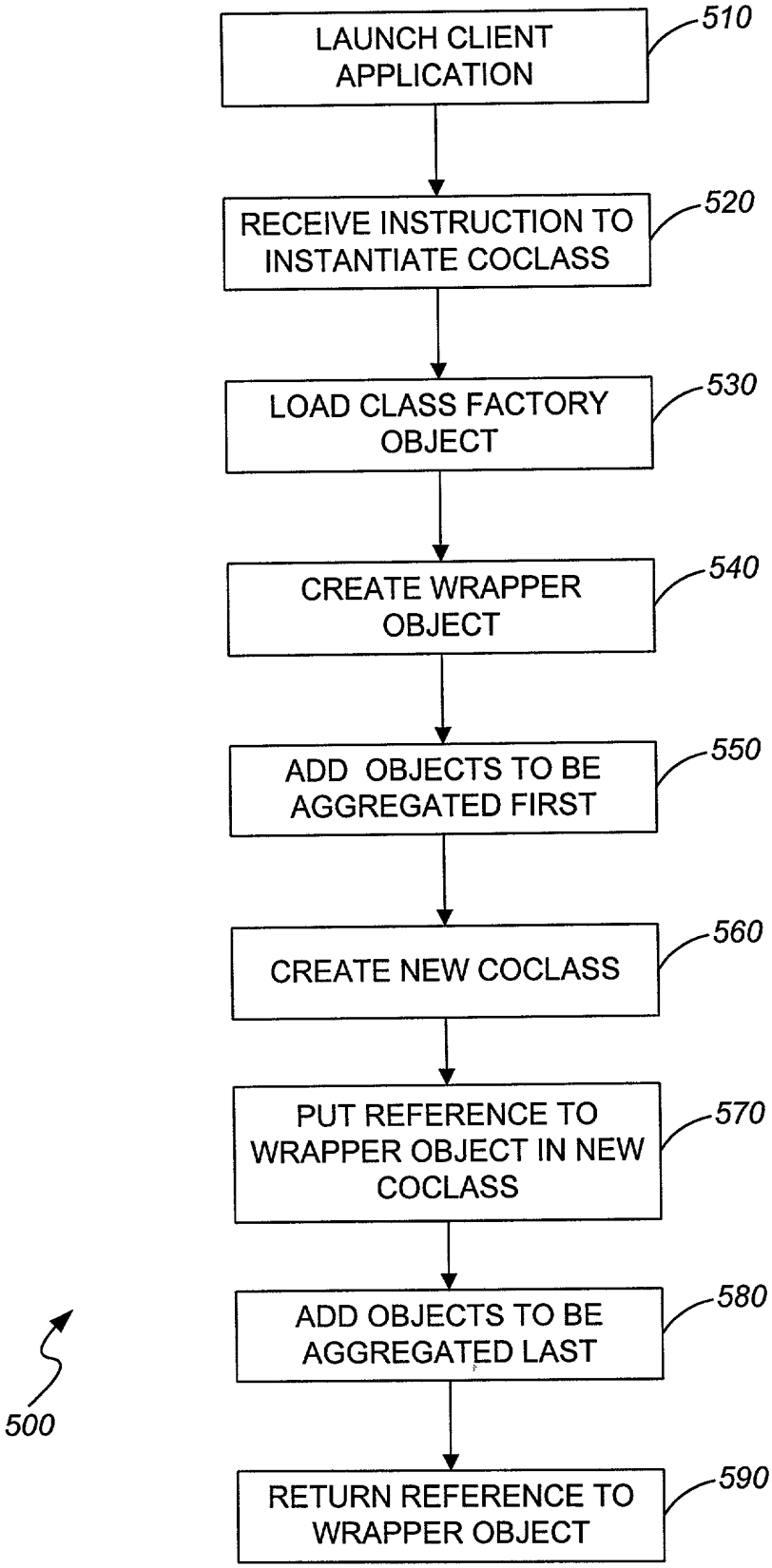

LAUNCH CLIENT APPLICATION
510
RECEIVE INSTRUCTION TO INSTANTIATE COCLASS
520
LOAD CLASS FACTORY OBJECT
530
CREATE WRAPPER OBJECT
540
ADD OBJECTS TO BE AGGREGATED FIRST
550
CREATE NEW COCLASS
560
PUT REFERENCE TO WRAPPER OBJECT IN NEW COCLASS
570
ADD OBJECTS TO BE AGGREGATED LAST
580
RETURN REFERENCE TO WRAPPER OBJECT
590
500

FIG. 5

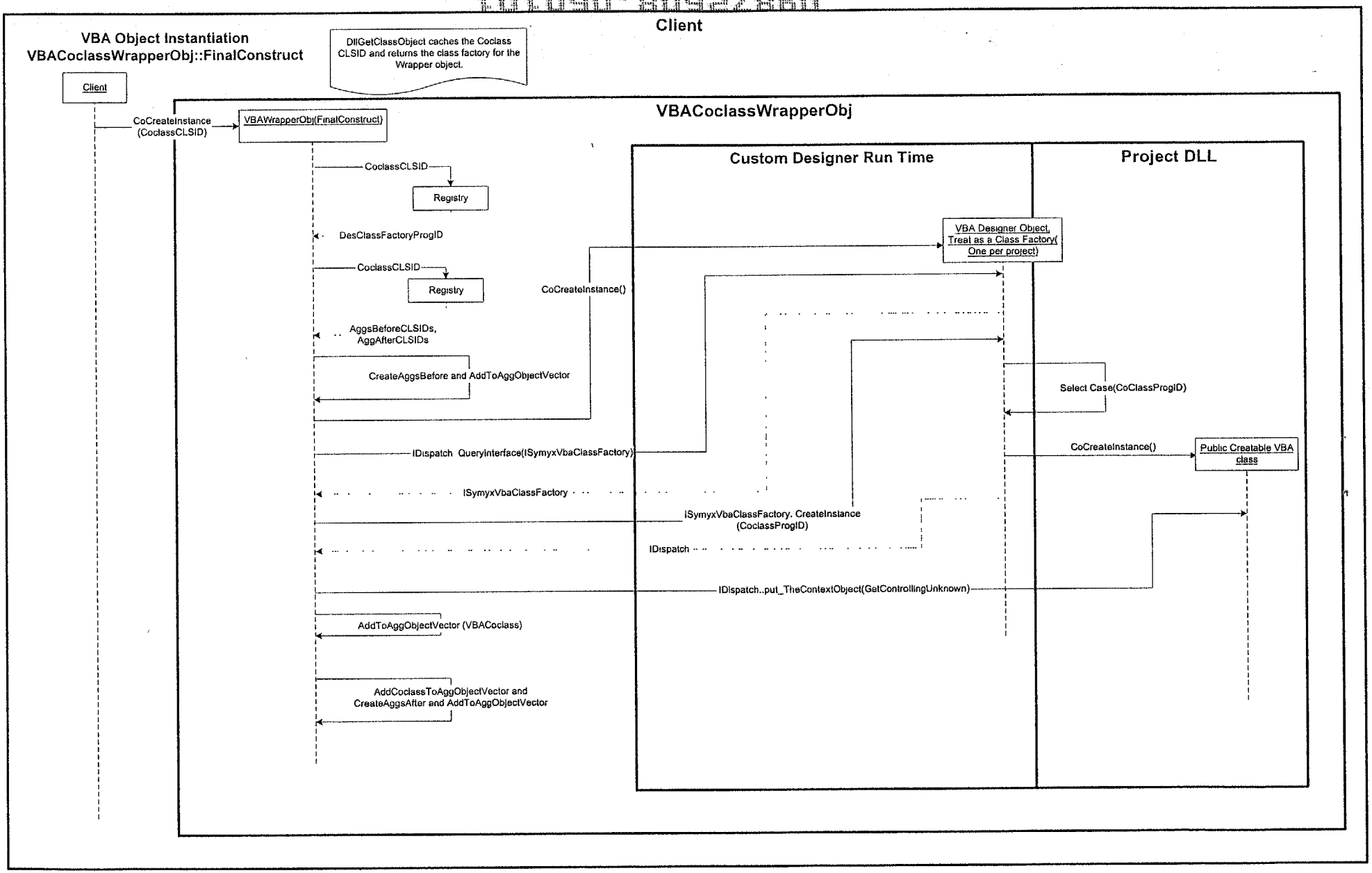

VBA Object Instantiation
VBACoclassWrapperObj::FinalConstruct
Client
DllGetClassObject caches the Coclass CLSID and returns the class factory for the Wrapper object.
Client
VBACoclassWrapperObj
Custom Designer Run Time
Project DLL
CoCreateInstance (CoclassCLSID)
VBAWrapperObj(FinalConstruct)
CoclassCLSID
Registry
DesClassFactoryProgID
CoclassCLSID
Registry
CoCreateInstance()
VBA Designer Object, Treat as a Class Factory(One per project)
AggsBeforeCLSIDs, AggAfterCLSIDs
CreateAggsBefore and AddToAggObjectVector
Select Case(CoClassProgID)
IDispatch QueryInterface(ISymyxVbaClassFactory)
CoCreateInstance()
Public Creatable VBA class
ISymyxVbaClassFactory
ISymyxVbaClassFactory. CreateInstance (CoclassProgID)
IDispatch
IDispatch..put_TheContextObject(GetControllingUnknown)
AddToAggObjectVector (VBACoclass)
AddCoclassToAggObjectVector and CreateAggsAfter and AddToAggObjectVector

FIG. 6

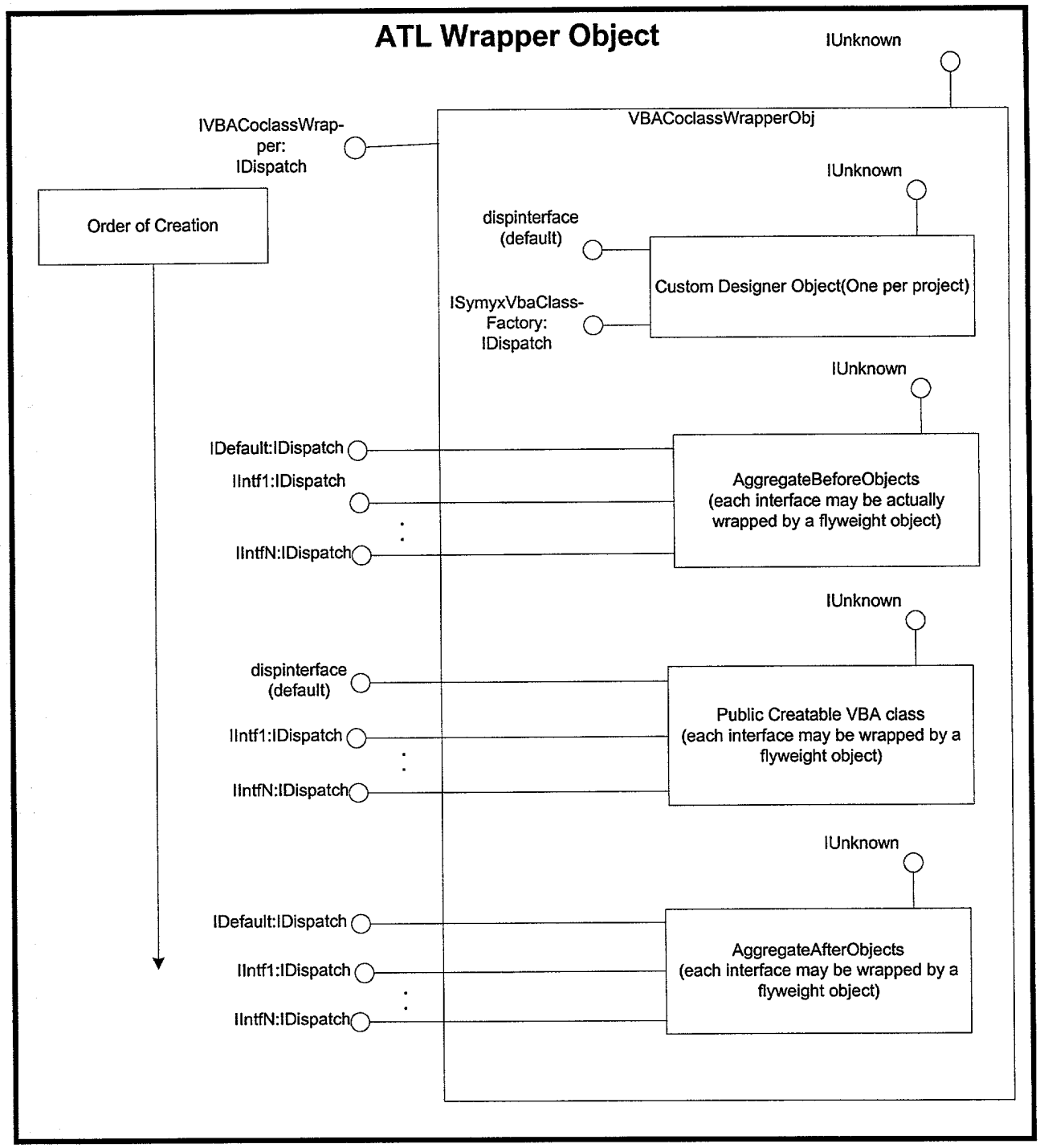

ATL Wrapper Object
IUnknown
VBACoclassWrapperObj
IVBACoclassWrap-
per:
IDispatch
IUnknown
Order of Creation
dispinterface
(default)
Custom Designer Object(One per project)
ISymyxVbaClass-
Factory:
IDispatch
IUnknown
IDefault:IDispatch
IIntf1:IDispatch
IIntfN:IDispatch
AggregateBeforeObjects
(each interface may be actually
wrapped by a flyweight object)
IUnknown
dispinterface
(default)
IIntf1:IDispatch
IIntfN:IDispatch
Public Creatable VBA class
(each interface may be wrapped by a
flyweight object)
IUnknown
IDefault:IDispatch
IIntf1:IDispatch
IIntfN:IDispatch
AggregateAfterObjects
(each interface may be wrapped by a
flyweight object)

FIG. 7

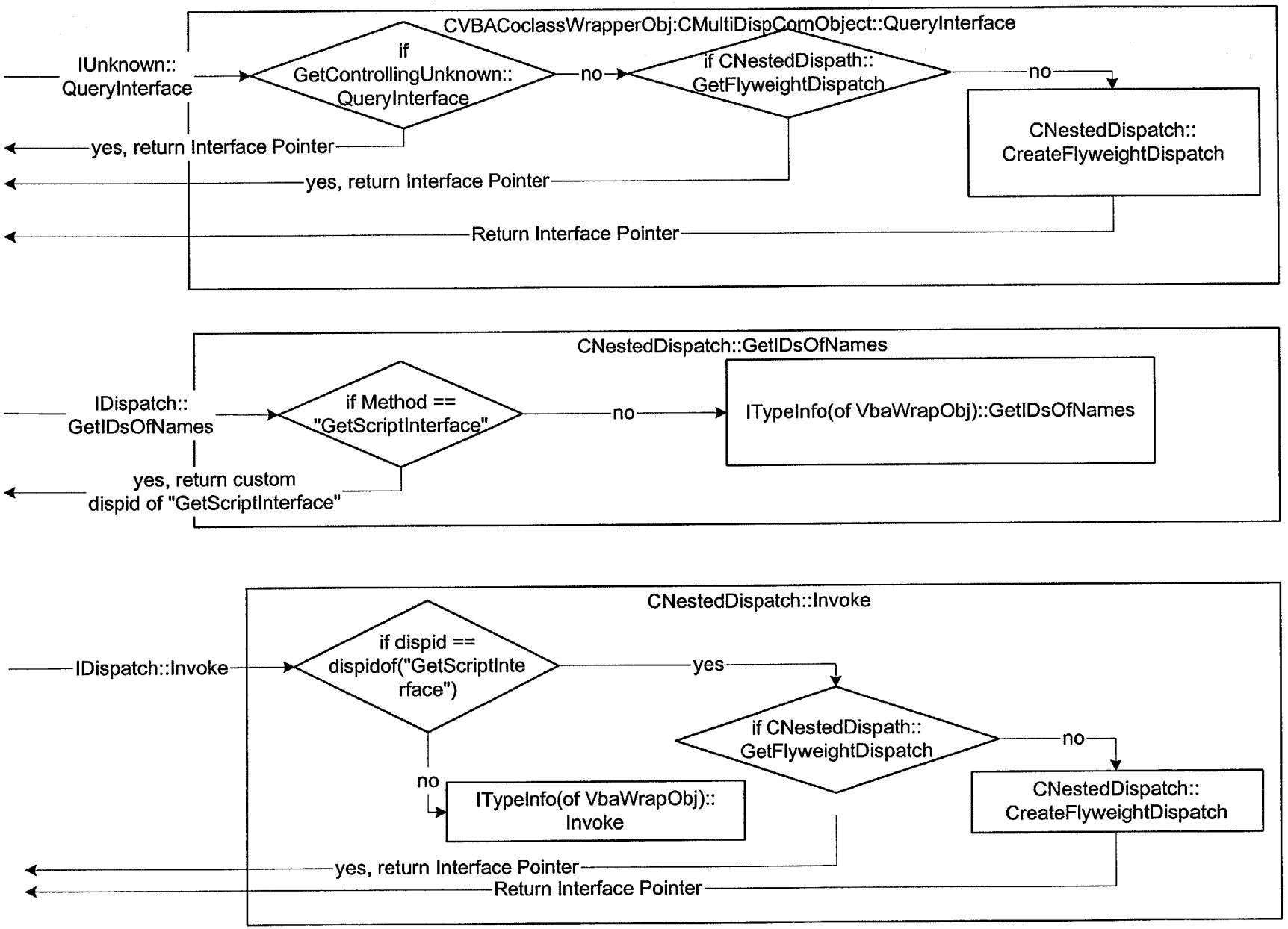
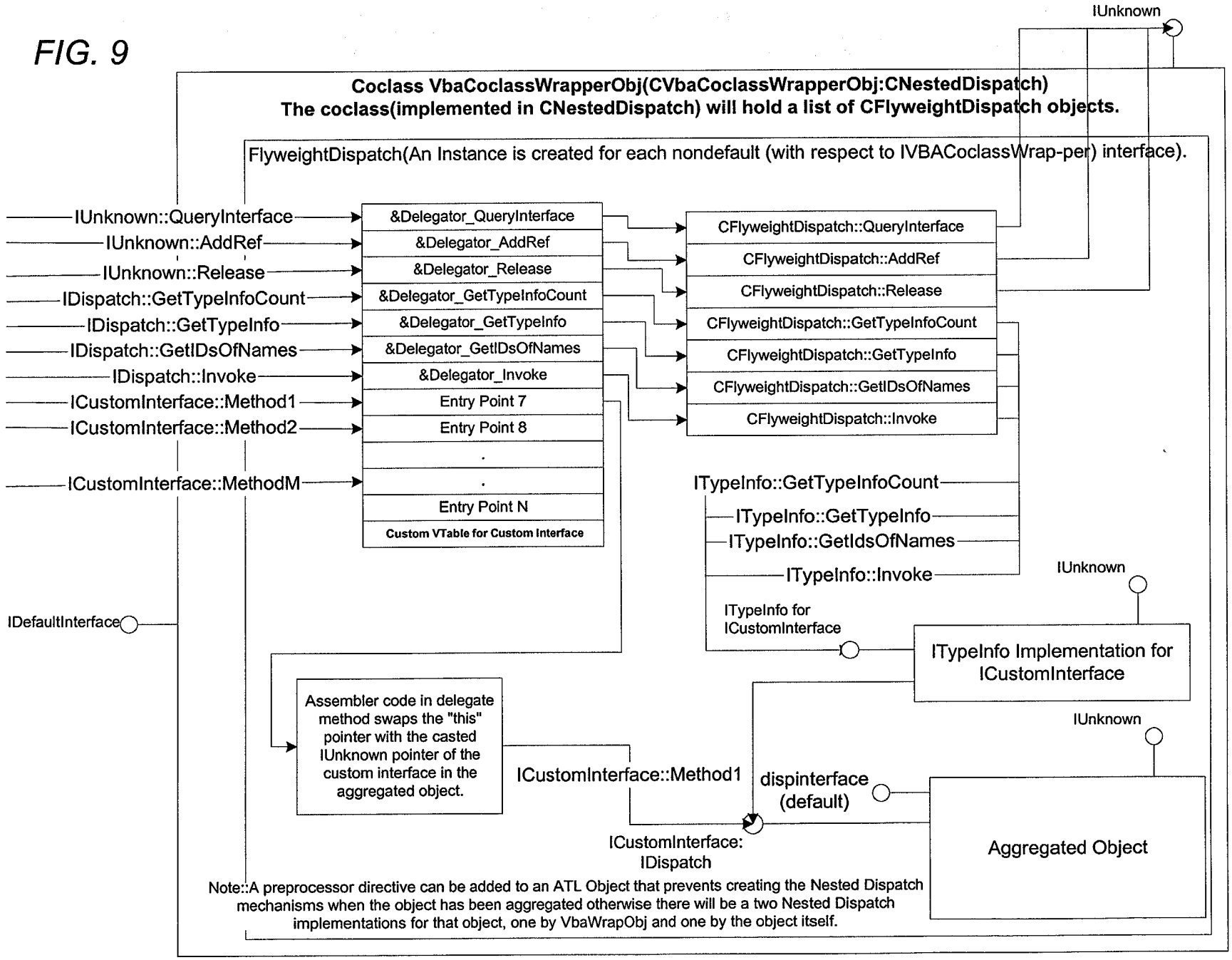

CVBACoclassWrapperObj:CMultiDispComObject::QueryInterface
IUnknown::
QueryInterface
if
GetControllingUnknown::
QueryInterface
no
if CNestedDispath::
GetFlyweightDispatch
no
CNestedDispatch::
CreateFlyweightDispatch
yes, return Interface Pointer
yes, return Interface Pointer
Return Interface Pointer
CNestedDispatch::GetIDsOfNames
IDispatch::
GetIDsOfNames
if Method ==
"GetScriptInterface"
no
ITypeInfo(of VbaWrapObj)::GetIDsOfNames
yes, return custom
dispid of "GetScriptInterface"
CNestedDispatch::Invoke
IDispatch::Invoke
if dispid ==
dispidof("GetScriptInte
rface")
yes
if CNestedDispath::
GetFlyweightDispatch
no
CNestedDispatch::
CreateFlyweightDispatch
no
ITypeInfo(of VbaWrapObj)::
Invoke
yes, return Interface Pointer
Return Interface Pointer

FIG. 8

FIG. 9


IUnknown
Coclass VbaCoclassWrapperObj(CVbaCoclassWrapperObj:CNestedDispatch)
The coclass(implemented in CNestedDispatch) will hold a list of CFlyweightDispatch objects.
FlyweightDispatch(An Instance is created for each nondefault (with respect to IVBACoclassWrap-per) interface).
IUnknown::QueryInterface
IUnknown::AddRef
IUnknown::Release
IDispatch::GetTypeInfoCount
IDispatch::GetTypeInfo
IDispatch::GetIDsOfNames
IDispatch::Invoke
ICustomInterface::Method1
ICustomInterface::Method2
ICustomInterface::MethodM
&Delegator_QueryInterface
&Delegator_AddRef
&Delegator_Release
&Delegator_GetTypeInfoCount
&Delegator_GetTypeInfo
&Delegator_GetIDsOfNames
&Delegator_Invoke
Entry Point 7
Entry Point 8
Entry Point N
Custom VTable for Custom Interface
CFlyweightDispatch::QueryInterface
CFlyweightDispatch::AddRef
CFlyweightDispatch::Release
CFlyweightDispatch::GetTypeInfoCount
CFlyweightDispatch::GetTypeInfo
CFlyweightDispatch::GetIDsOfNames
CFlyweightDispatch::Invoke
ITypeInfo::GetTypeInfoCount
ITypeInfo::GetTypeInfo
ITypeInfo::GetIdsOfNames
ITypeInfo::Invoke
IUnknown
ITypeInfo for ICustomInterface
ITypeInfo Implementation for ICustomInterface
IDefaultInterface
Assembler code in delegate method swaps the "this" pointer with the casted IUnknown pointer of the custom interface in the aggregated object.
ICustomInterface::Method1
dispinterface (default)
IUnknown
ICustomInterface: IDispatch
Aggregated Object
Note::A preprocessor directive can be added to an ATL Object that prevents creating the Nested Dispatch mechanisms when the object has been aggregated otherwise there will be a two Nested Dispatch implementations for that object, one by VbaWrapObj and one by the object itself.

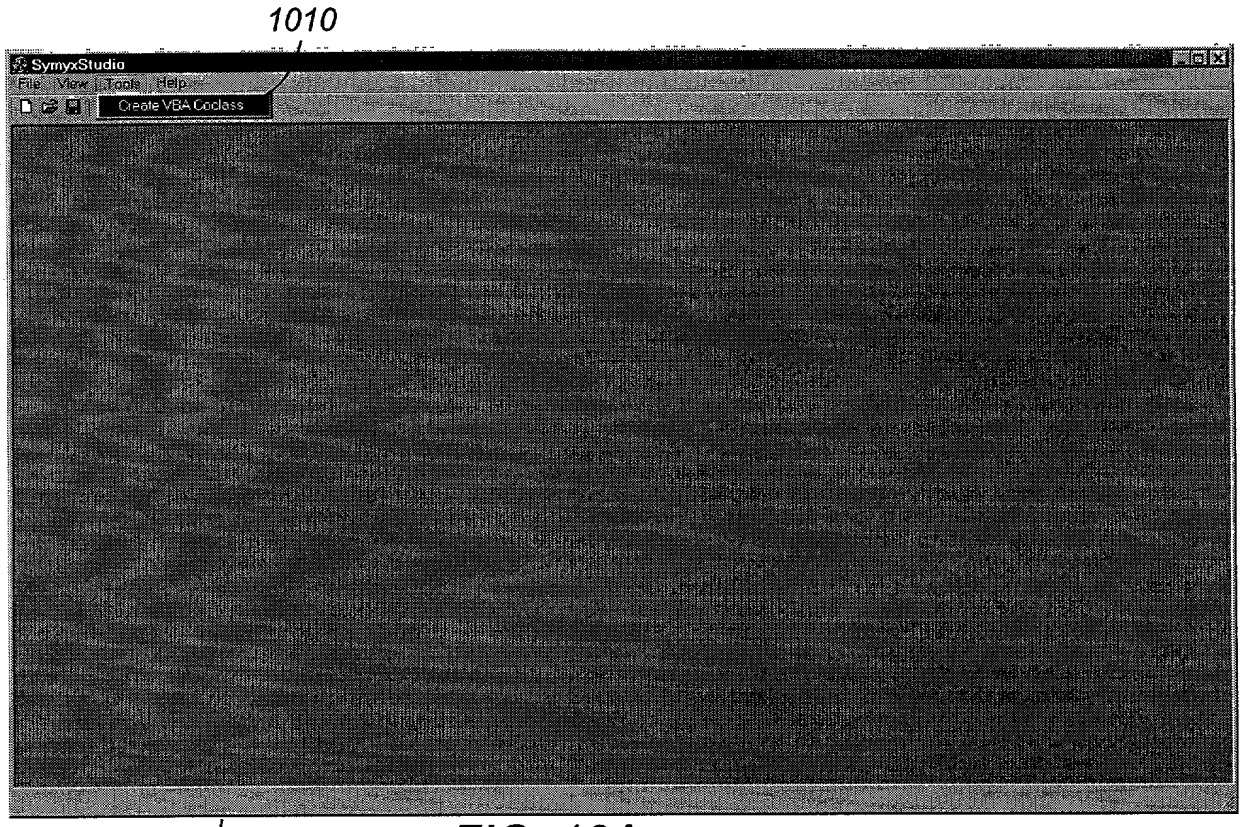


FIG. 10A

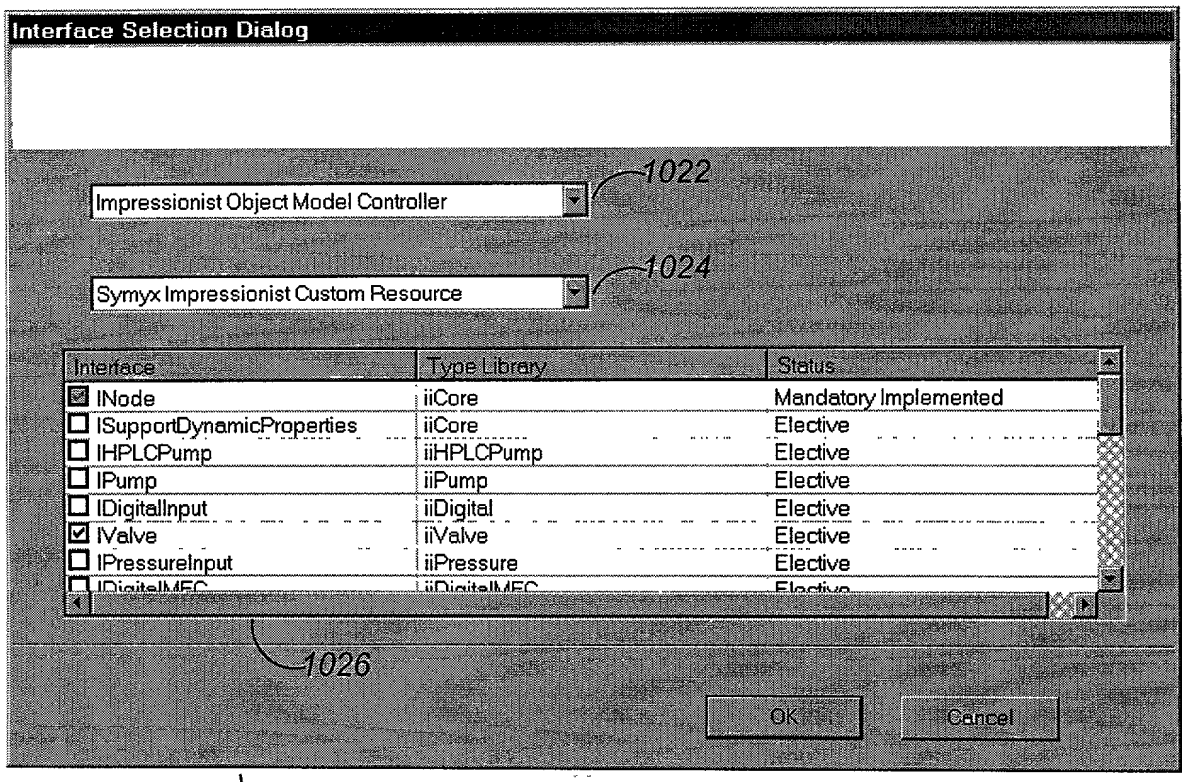


FIG. 10B

US 6,826,568 B2

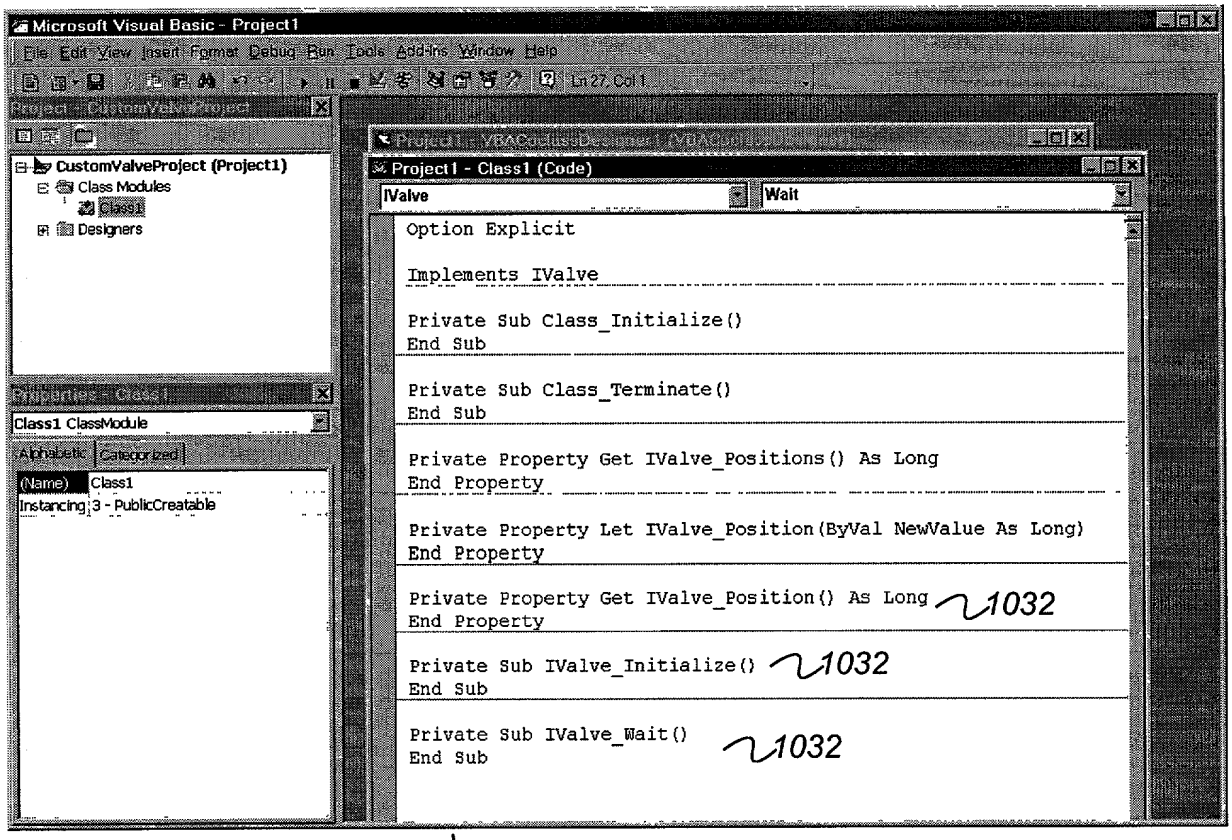

Microsoft Visual Basic - Project1
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 27, Col 1
CustomValveProject (Project1)
Class Modules
Class1
Designers
Class1 ClassModule
Alphabetic
Categorized
(Name) Class1
Instancing 3 - PublicCreatable
Project1 - Class1 (Code)
IValve
Wait
Option Explicit
Implements IValve
Private Sub Class_Initialize()
End Sub
Private Sub Class_Terminate()
End Sub
Private Property Get IValve_Positions() As Long
End Property
Private Property Let IValve_Position(ByVal NewValue As Long)
End Property
Private Property Get IValve_Position() As Long
End Property
Private Sub IValve_Initialize()
End Sub
Private Sub IValve_Wait()
End Sub
1032
1032
1032
1030

FIG. 10C

Patent Application Publication Aug. 30, 2007 Sheet 10 of 10 US 2007/0204203 A1

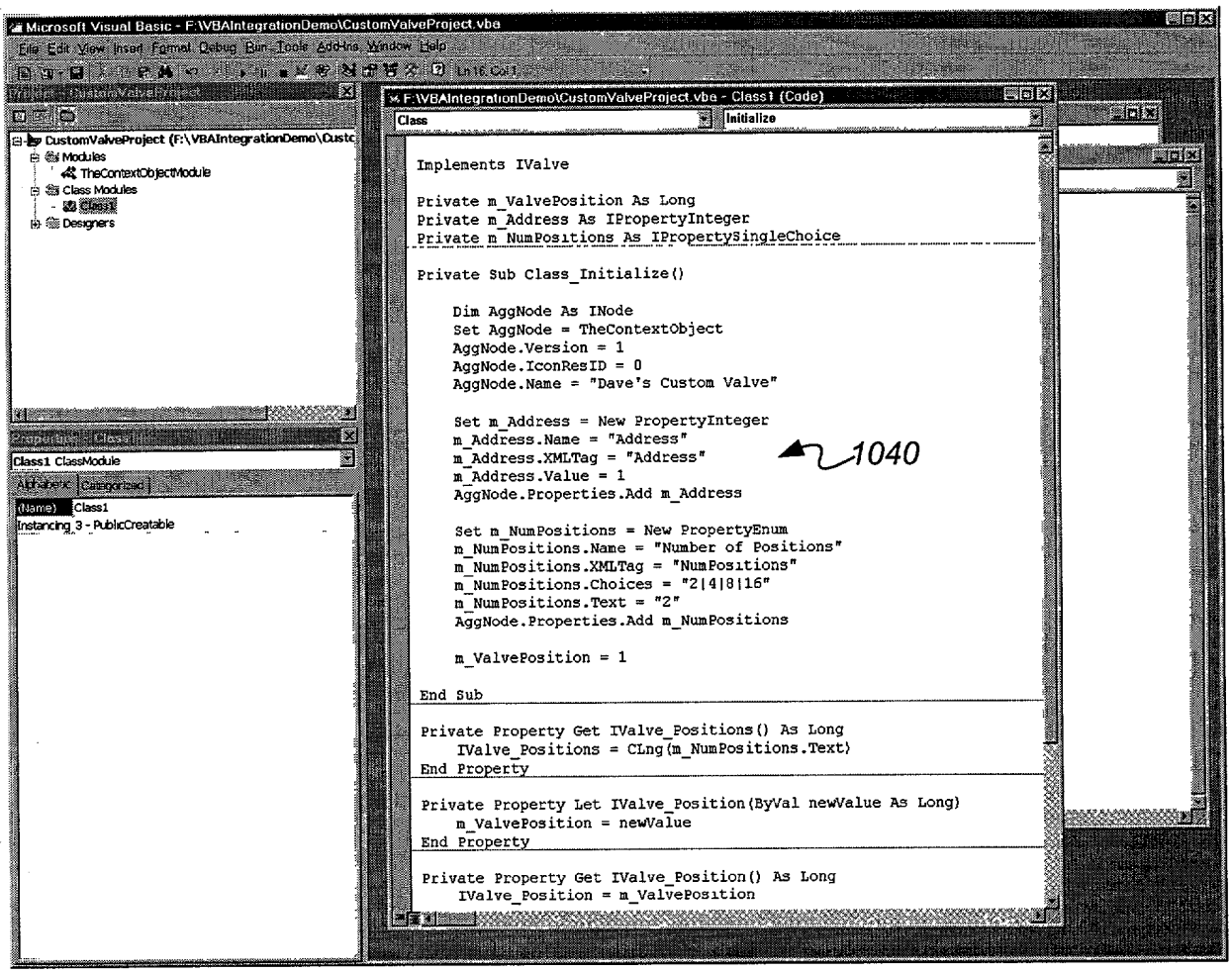


FIG. 10D

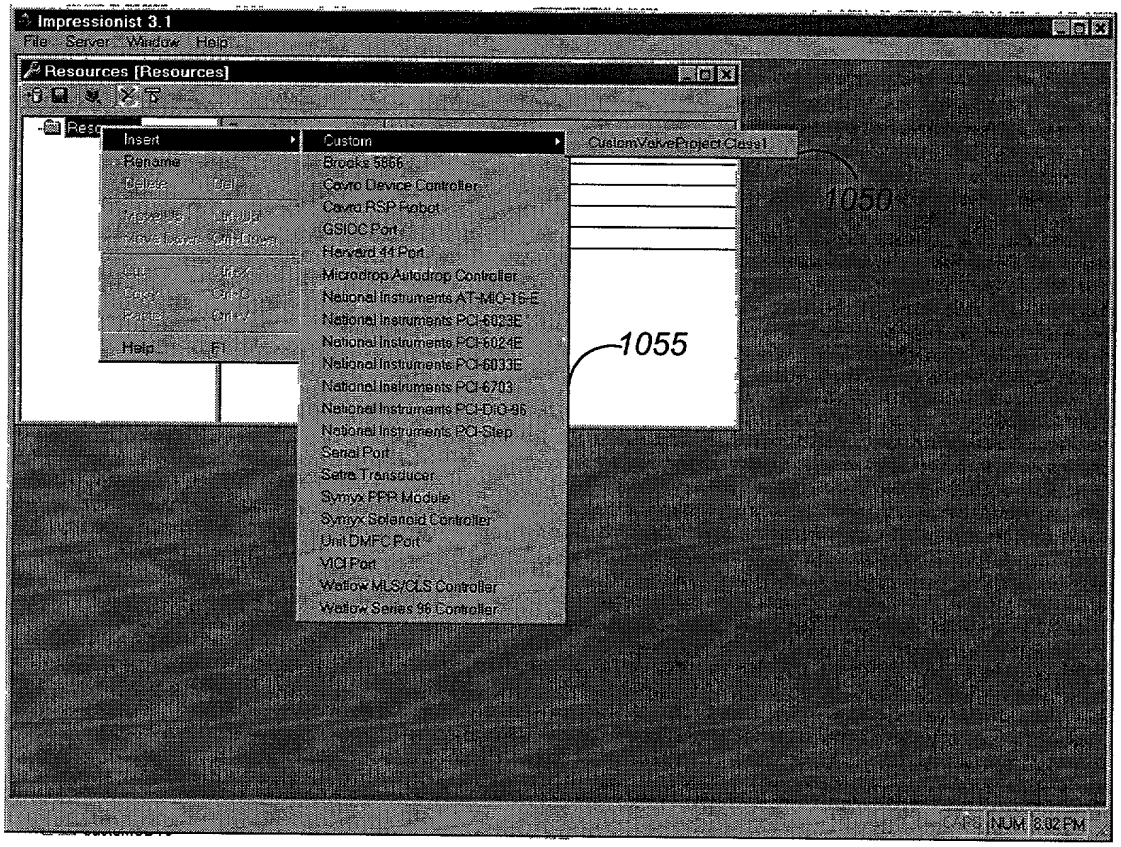


FIG. 10E

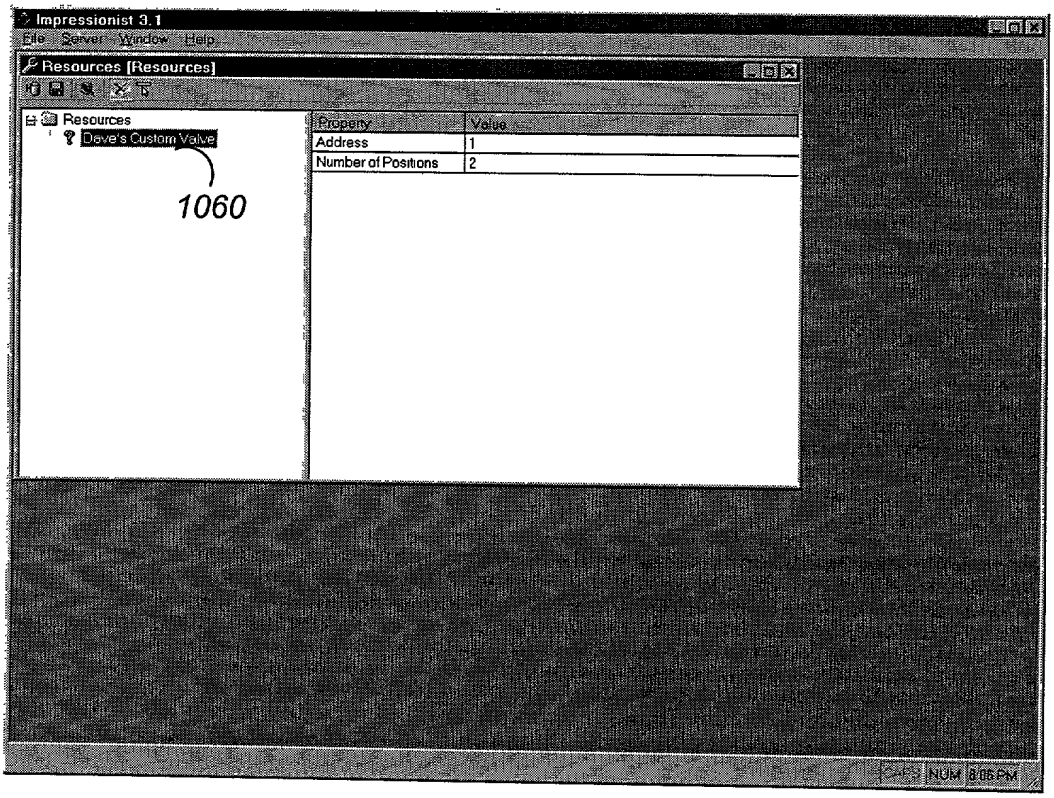


FIG. 10F